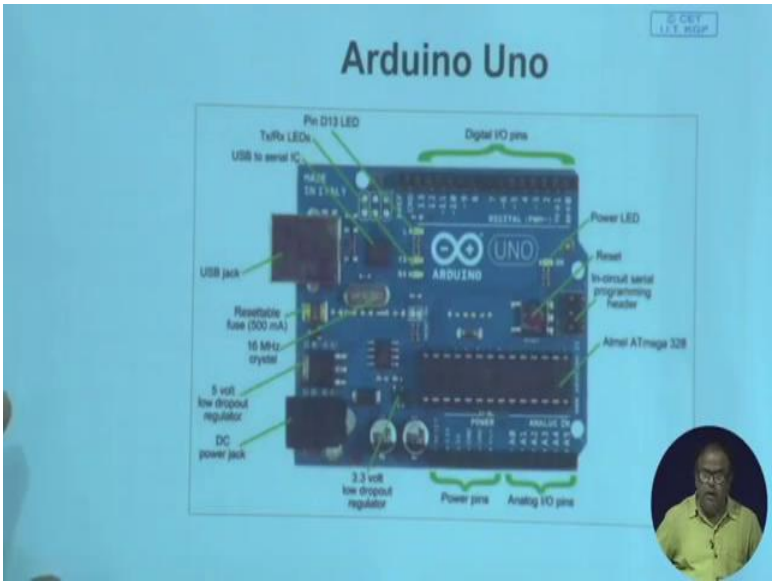
Embedded Systems Design
Prof. Anupam Basu
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 16
Arduino Uno

Today, we will start looking at category of processors.

(Refer Slide Time: 00:23)

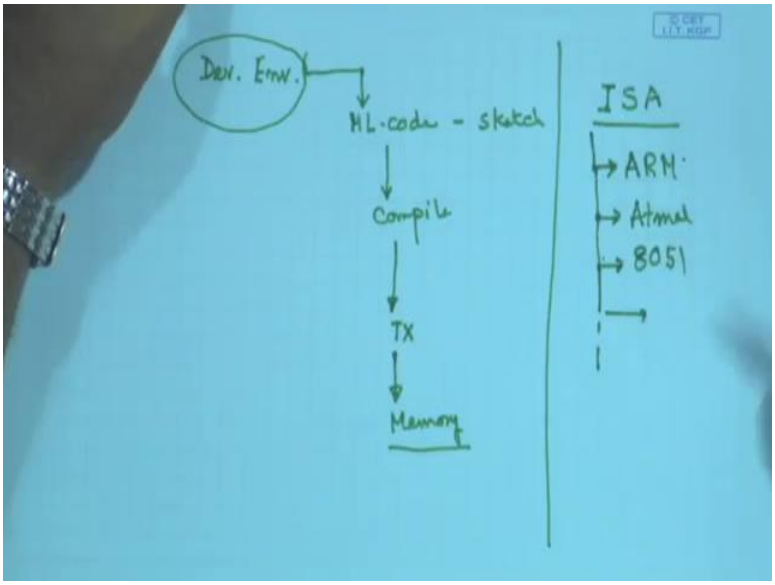


We have seen earlier in our discussions, so we have talked about FPGA's. So, FPGA's are field programmable database and we have seen Zylinks and other FPGA's. Here I am introducing to you the Arduino group of controller; micro controller volts just like Zylinks; the FPGA's and Zylinks at the core and there were other devices mounted on the same FPGA board. Similarly, Arduino board has got at its core a micro controller; for example, when you talk of Arduino Uno; that is the particular board which has got Atmel AT mega 328.

So, it is a 32 bit micro controller and on this board we have got several things like we have got a number of digital I O pins, there are some analog output pins also here and there are analog I O pins, there are power pins, there is a reset there LED's, power LED and there are transmitter receivers, I have got the interfaces like the USB, DC power jack everything is ready for you.

So, we are discussing about a micro controller board of the category Arduino, which has got it as it is core the micro controller which is Atmel; this Atmels you can also have there are Arduino duo or duets which has got ARM as the micro controller or micro processor of the core, so there can be different boards; now along with these boards come some development environment.

(Refer Slide Time: 03:01)



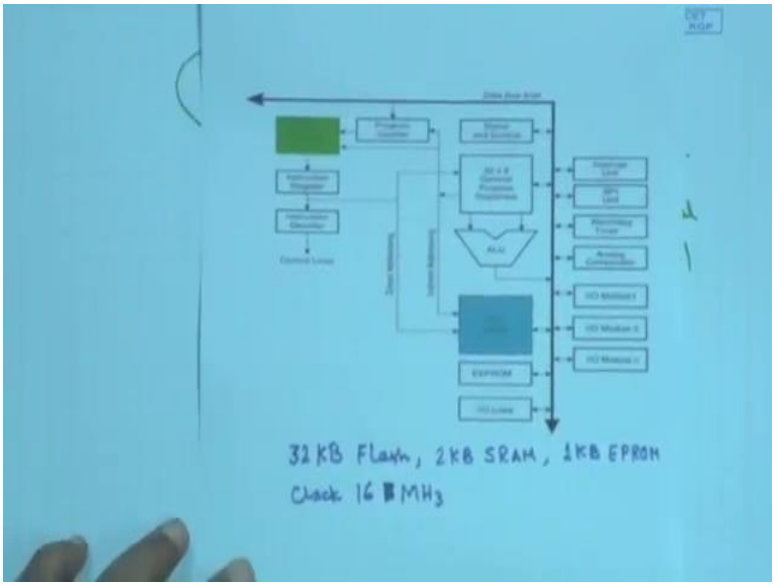
So, actually what happens is; there is a development environment and using the development environment you write the code; sort of high level code which is known as a sketch, sketch of what you want to do alright, then you compile it.

Ultimately we have to learn in the machine level machine code, so that one can be transferred using the transmit connectivity to the memory of the micro controller, so that is the process. So, we have to write at this level for development purpose you need not write at the machine code, but I would very much recommend you and encourage you, I think I have told that in an earlier lecture; I do not recall it very much now, that when I was discussing about micro processors and micro controllers as a general purpose processors, I said that they are varying in the Instruction Set Architecture; ISA. So, I very much recommend you to look at the instruction sets of different processors like ARM, like Atmel; like Atmel is basically very close to 8051 micro controller, there are many others.

So, you must read them and see how the instruction set varies; depending on the instruction set, the power of the micro controller varies. What are the things can that can be done using efficient coding, the code efficiency of course for all of these micro processors since their general purpose processors; I can practically do everything with these, but how efficiently can I do this that will very much depend on the instruction set architecture of this. So, here we have given some high level instructions with which we can write the program which is also called the sketch and that will be compiled and it will go to the memory.

So, now we will first discuss about; so once again let us repeat what we have here, we have got the micro controller positioned here and there is a clock of 16 megahertz; a crystal here and there are analog; analog I O's and digital I O's and power pins, there are transmission transmitter receiver LED's also.

(Refer Slide Time: 06:22)

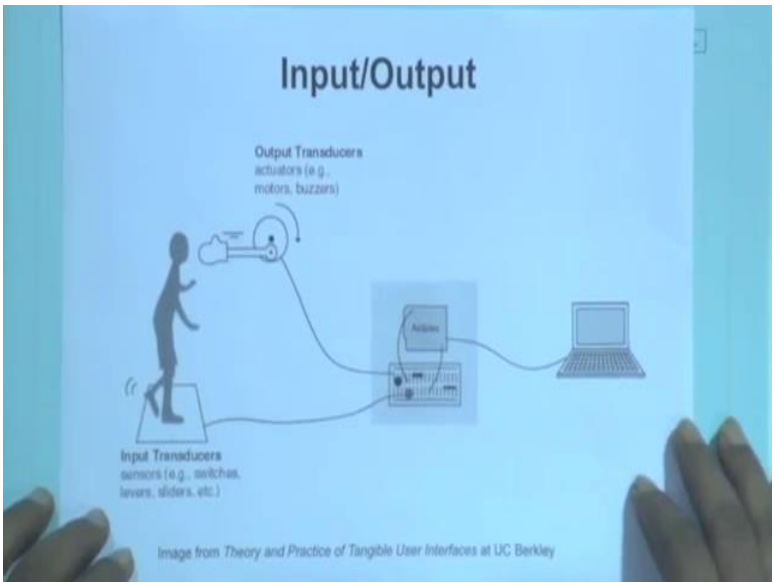


Now just like other micro processors, we can have a quick look at the Atmel micro processor; I do not know how much is visible one of the very important features of this Atmel; which is housed in Arduino is that it is of hardware architecture. I hope you recall what is hardware architecture, the program memory and data memory has separate; here we have got a flash program memory shown in green here and a data memory here which is S RAM based alright.

We have got an ALU inside the; this is the micro controller, not a micro processor alone. Therefore, I have got like a normal micro processor I have got the instruction register, instruction decoder, program counters all those things are there, in addition I have got the e e prom; 1 kilobyte of e prom is here, 2 kilobytes of S RAM is here and 32 kilobytes of flash memory is here; there your KB and here I have got an interrupt unit, I have got watchdog timer which is very important, serial port interface, I have got analog comparator and several I O modules; that is inside that particular chip which is Atmel and there it is clocked at the rate of 16 megahertz.

Now, there are other versions of Atmel; this is the Atmel mega, but there are other versions of Atmel which turned at much lower clock speed, there are much simpler; I mean less powerful, but we have just taken this as an example. So, once again; so what we have here is the Arduino board with the Atmel here and that Atmel is what we have just now shown here; that is the Atmel. Now we have to do something with this, we have to achieve something with this.

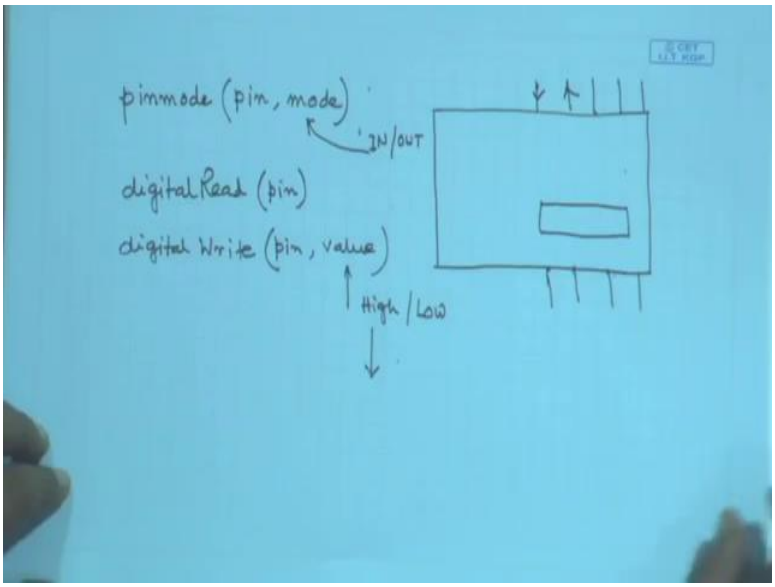
(Refer Slide Time: 08:58)



So, the typical way that we go about is we need to do I O. So, you can see this that there can be input transducers; input transducers can be in any form say for example, this is a nice thing that when somebody comes inside the room, there will be some light automatically glowing. So, what we need to do is an input transducer which may be pressure sensitive, special sensor and that sensor is sensed by the Arduino board and the

Arduino board automatically activates it is on actuator which may be an LED, which may be a buzzer or which may be anything and it maybe also connected to the computers., so that we can record the events.

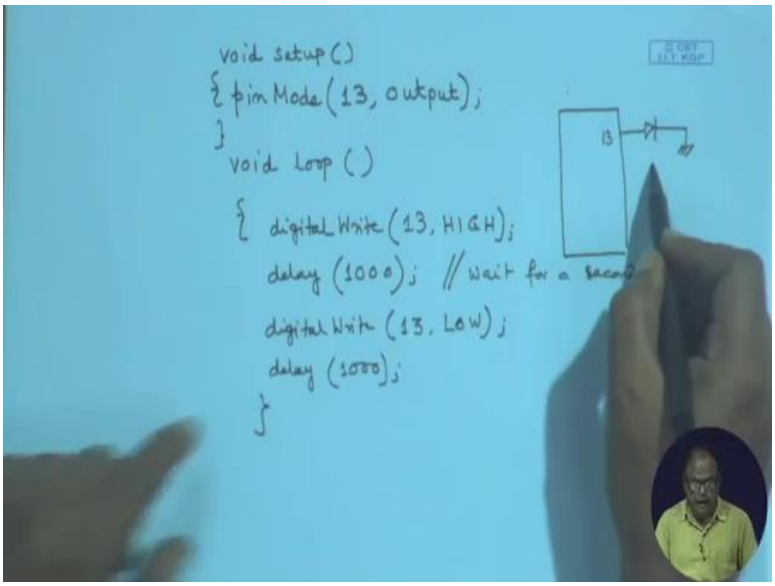
(Refer Slide Time: 10:21)



Now as I said that you have to write the sketch, so for that the typical way we write there are some very useful commands given. You see the Arduino board is something like this, we have got the processor here and we have got the number of pins all around; just as we had with in an (Refer Time: 10:35) FPGA. So, we have got the command pin mode, where I can identify a pin number and I can specify the mode, whether it is input or output. So, this mode will be either input in or out. Similarly so any of the pins here, I can set them in the input board or in the output board.

Similarly, I have got digital read a particular pin, now if I write digital read, now you know that some pins are digital I O, some are analog I O I like to specify one of the digital I O pins with digital read; that means, that pin must be in the input mode and then I can read the digital output. Similarly I can have digital write on a particular pin and I can put a value because I am writing something out clear. So, I write digital, so this value will be high or low; here of course, I am not writing some decimal value, it is either high or low and here also digital read; it reads either high or low. So, typically high or low means what, if I am putting out then I am actually driving some current in; when I am doing in then I am sinking some current in, so that is how we carry out the I O here.

(Refer Slide Time: 13:01)



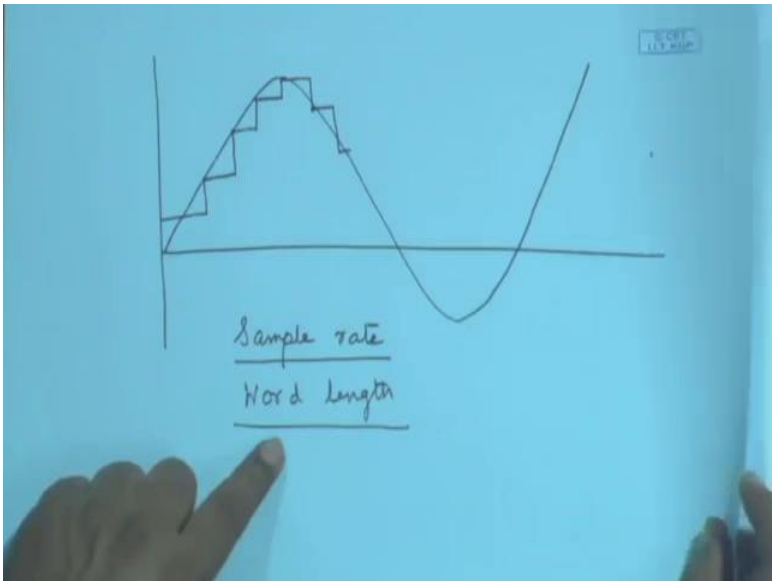
Now, let us start writing a simple program say, so I write pin mode some pin 13 to be output; this is a digital pin. Now before that I have got just like C++, now by the way one thing I forgot to say is that these sketches are written in C++ type. So, I can have a set up like void, I am writing a setup function setup something and then I start alright just like C++. So, pin 13; suppose here is my Arduino and pin 13 has got an LED connected with this alright; some LED is connected with this. So, actually in most of the Arduino boards; an led is connected to pin thirteen. So, if you look at this you will see that in some pins like this 13 LED's are automatically put in there alright here. So, that will automatically glow if I write a 1 there, but how do I write a 1, so here I will do void loop then.

Student: But we have to do (Refer Time: 14:51) setup then.

We have done the setup and the setup will be closed of course here and in void loop digital write 13, HIGH, then delay 1000; this delay 1000 will take 1000 millisecond; this values in millisecond that is wait for a second alright. So, I give a delay and then I again write digital write 13 low; delay 1000 and this loop continues. So, what will happen this LED will glow for 1 second and again will be off for 1 second, again will glow for1 second again will be off for 1 second. Similar programming we have done for with FPGA, where we wrote the same thing in VHDL and dumped the big pattern on the FPGA.

Here once you do this, we can compile it in the Arduino I d e; the development environment and we will get in the big pattern, so far for the digital inputs.

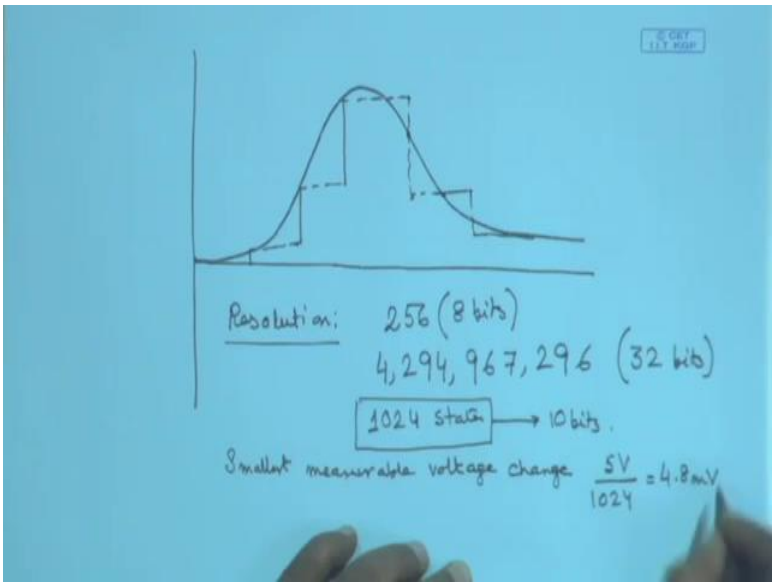
(Refer Slide Time: 17:00)



Now for the analog inputs say we know that, we will have some analog signal, once again I repeat that I have got some analog signal, but I am not reading that analog signal straight way I am; something this is not very symmetric though, but I am sampling them; so I am getting some sample values so and so forth, again I go on sampling. Now as I sample I have got two important criteria to think about; one is the sample rate which you have already discussed that it must be twice the highest frequency of this signal and accordingly as the sample rate is increasing will be challenged with a word; word length.

If I increase the sampling rate then the number of bits to encode because there will be more number of samples, so I have to think of the word length that is another point that you have to think about.

(Refer Slide Time: 08:15)



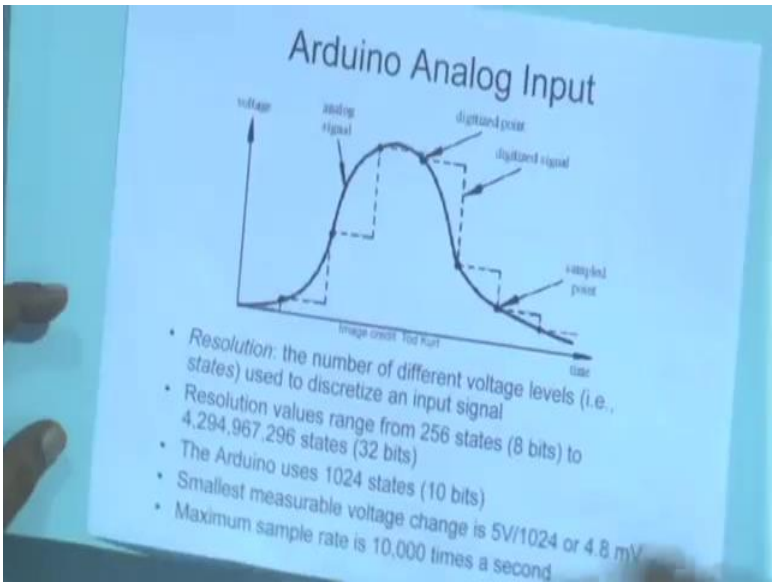
Now, let us take another feature about suppose I have got a signal like this alright and I have sampled it here, then I have sampled it here and I have sampled it here etcetera and I have sampled it here, here and then sampled it same rate of sample it here or some are some are like this. So, now I have got to bother about the resolution, just to refresh your mind; what is resolution? Resolution is the number of different voltage levels used to discretize an input signal.

How many voltage levels; I discretize the input signal in how many voltage level that is my resolution. Now the resolution values range from; if I have got 8 bits, then it will be 256 and if I have got 32 bits, it will be very huge something like this for 32 bits. So, this is a huge resolution. Now the Arduino, we will be using 1024 states for sampling; 1024 states means it will be 10 bits, so we will be using 10 bits; so 2 to the power 10. Now, therefore the smallest measurable voltage change is how much? What is the smallest if 5 volt d max; 0 to 5 volt is my swing, so what is the smallest measurable voltage change?

Student: 5 volt.

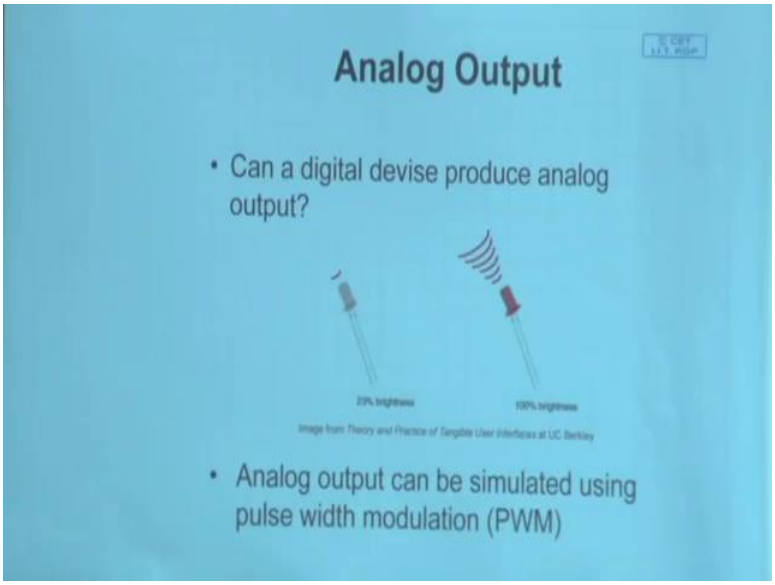
5 volt by 1024 that will be around 4.8 milli volt and the maximum sample rate will be 10 thousand per second because I have got 10 bits, so, that is the specification.

(Refer Slide Time: 21:26)



So similar to digital inputs; I have got analog link and analog light, we will see that through some example.

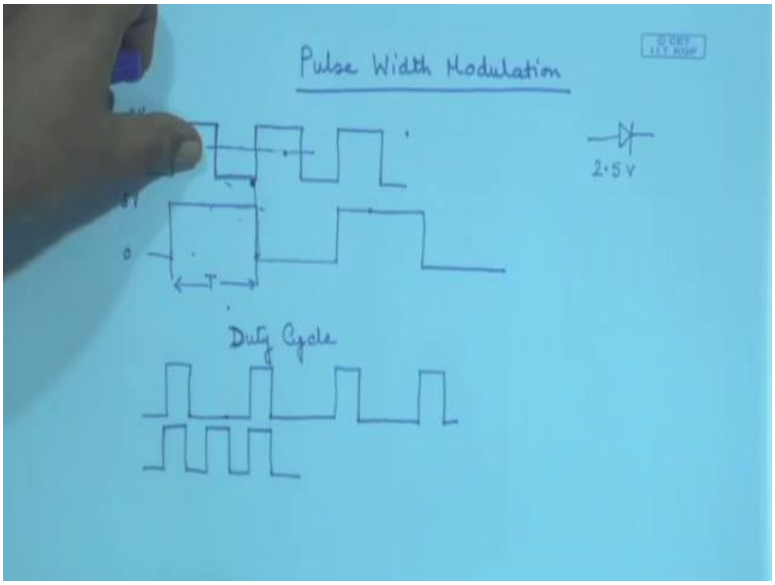
(Refer Slide Time: 21:44)



Now, next question that we would like to see is that suppose I have got an LED; I want to of course, with this LED is connected to pin 13. So, I can make it on or off through digital output; digital write, but if I want to make it glow in a continuous form that mean say with 23 percent brightness for some input, 100 percent brightness for some input; how can we do that? So, first of all we are talking about the analog output, how can we

change the analog output in this? Now can a digital device produce analog output? Yes or no; obviously, the answer is directly it cannot do.

(Refer Slide Time: 23:14)



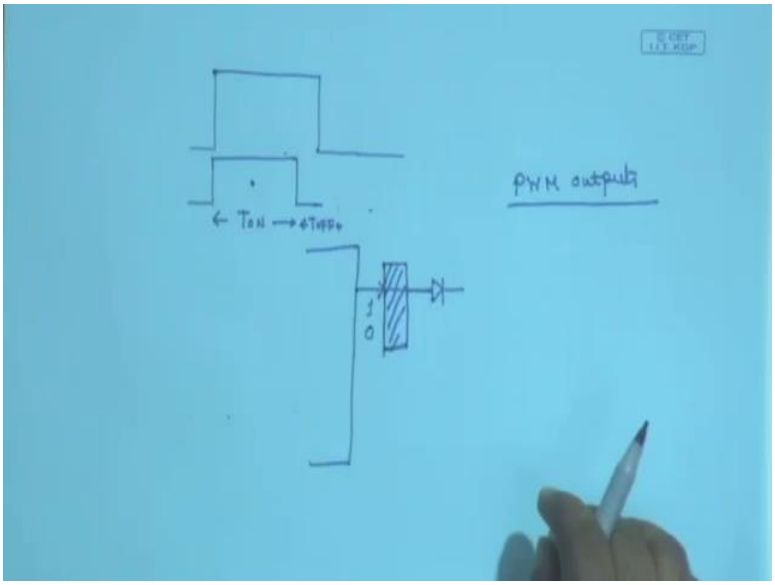
But we can play a trick for that; we have got a thing called pulse width modulation, some of you might be familiar with this. Suppose I want to say that LED that was there, I want to supply 2.5 volt; I cannot directly to the digital pin I cannot send 2.5 volt.

But suppose if my pulse is like this; compare these two waveforms; say here 0 to 5 volt. Now during this time, if I take this part; during this time I am getting 5 volt all through; however, if this will be my total period then here 50 percent of this time, I have kept it on and 50 percent of my time; I have kept it off. Therefore over this period, the average value of the voltage that is going through this is 2.5 volt; half of that I am sorry not up to this here. So, I can by controlling this on and off periods.

So, this is the duty cycle they are two important terms for this; one is the duty cycle, what percentage of the time is it on for a period? So, I can see the duty cycle here is 50 percent; it is 100 percent. So, for 50 percent of the time, the duty is being given the remaining 50 percent of the time it is sleeping that is how I think, now if I change this to a; so my total thing is up to this, if I draw it like this then; obviously, compared to this one the supply; the average voltage there will go out will be less here; if I do it much faster assuming the same amplitude, this is the period; I can change the period. So, there

can be two ways I can control the voltage; one is by changing the frequency of the one periods, the other period is by changing the width; How long it will it be on?

(Refer Slide Time: 27:31)

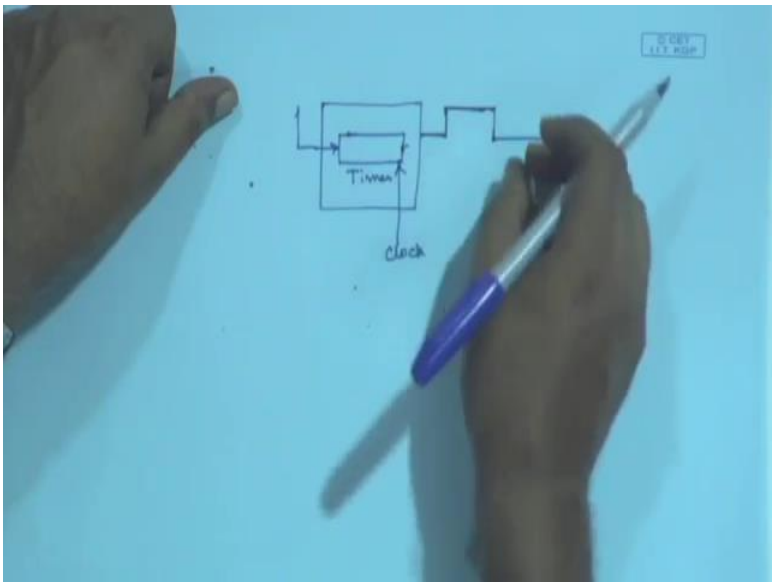


For example let me just do it once again, so what I do is whenever I want higher voltage then my original thing is this, so the supply is like this. Now I change the width of the on time and the off time, so this T on and this is T off; by modulating, by changing this I can control the average voltage over here. Therefore, although I have got a digital output say here is my volt, although I have got a digital output; if I can control of course, if I want to control say I cannot do it like this; say I have got this, I will connect something here, I will connect something here; What would I connect? That what will this thing do; that if I put it one, then this par, this current will go here, but then if I put it 0 then it will change.

So, you are right that I can apply here digital to analog convertor, but what that is not needed in this case because Arduino allows us to have some PWM outputs there is some PWM output pins.

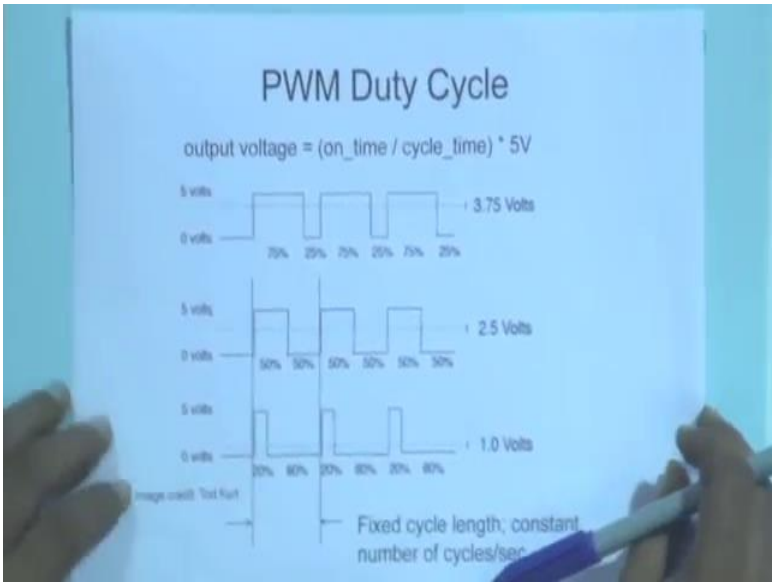
Suppose those pins were not there, what would I do? Suppose that pins are not there, how would I implement this if I do not have that PWM output? What I will do?

(Refer Slide Time: 29:51)



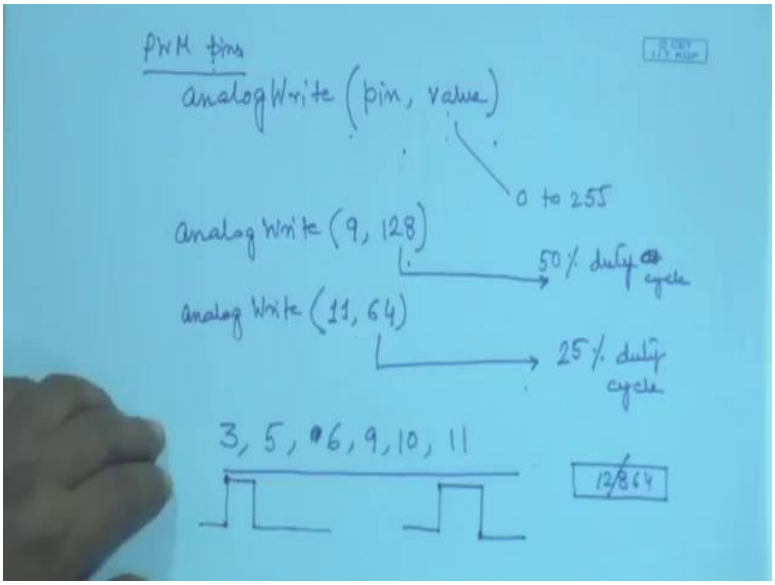
I will simply have whenever I am having on I want a period. So, I will set some circuit to be on for a particular duration, so I put it on and then I can have the timer here, this is a timer which I load with a particular on time. So, it will be on with a clock and this it will be it will be set to on; this will be set at clock alright and this value will be on; at the clock this is on. Now this timer will count down and soon as this time is over, then it will switch off this circuit. So by changing this timer; I could have controlled the on time and off time; however, this over it is not required in Arduino because we are having the PWM pins available.

(Refer Slide Time: 31:18)



So, here once again let us look at this I think this is visible; the output voltage that will be getting ultimately is on time; this on time by the cycle time, the cycle time is 100; times 5 volt. So, here you can see 75 percent of the time is on and then it is off, so the output voltage of this will be 75 percent of 5 volt; 0.75 into 5. Here this is a 50 percent duty cycle, so it will be 2.5 volts; here it is 20 percent duty cycle, so it will be 1 volt.

(Refer Slide Time: 32:21)



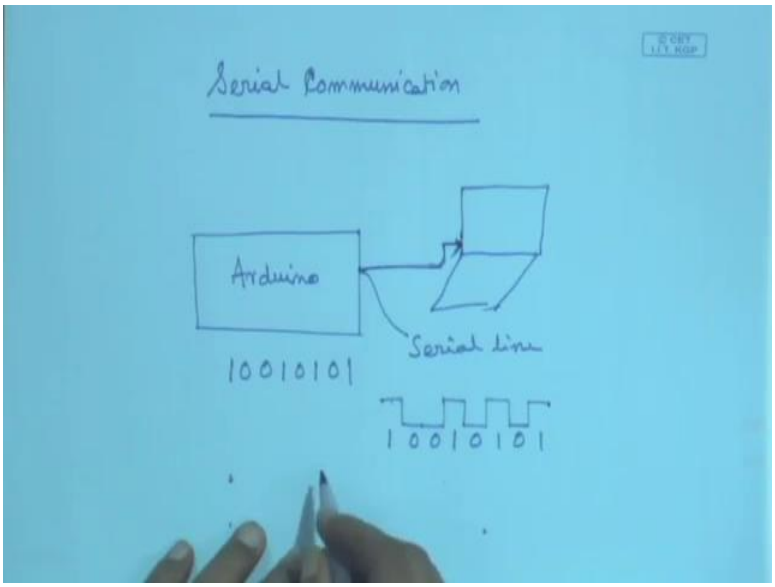
So, in Arduino we have got the command analog write; the pin and the value. The value is the duty cycle for the PWM pins; for the PWM pins we will write analog write and this value is between 0 to 255; that means if I write analog write pin number 9, 128; that means, pin number 9 will be fade with the pulse with 50 percent duty cycle.

If I write analog write say pin number 11, 64 then it will be 25 percent duty cycle. Therefore, you can very well imagine that the pins 3, 5, 6, 9, 10 and 11 are the PWM pins; these are got built in PWM circuits, so that they can generate PWM. So, by this you can understand that from here I can connect to multiple outputs and can provide multiple drives, multiple voltages to multiple different devices using PWM. Of course, I may do it in other ways also, but wherever I can manage it with PWM that will be very handy for Arduino board and here you must have understood what the circuit otherwise you would have to do if you want to generate it; that means, you have got this full duty cycles, so whenever you get the starting pulse, you start the timer and put this value 255 or 128, so what you are doing actually you are putting the timer with this 128.

So 128 in binary, so it starts off in this way I will start counting down; 8 bits, so here put 128. So, it will come down at every clock, so at fifty percent of the time; after 50 percent, it will become 0; if I put it to 64; the timer be; it would be on, the timer start immediately counting down 64. So, it will come out here, but that you need not do that circuit is already in built; that is the very big advantage in this development system.

The other thing that we have to know there are two other things that we have to know about Arduino, one is the serial the communications; how do we do serial communication?

(Refer Slide Time: 36:22)



The serial communication whenever I have got the Arduino board here and I want to connect to my PC; I like to connect to this. Now this I can do through a serial port, serial line and how is that serial; suppose some data pattern 1 0 0 1 0 1 0 etcetera 1, 2, 3, 4, 1; I want to send, then in the serial mode it will send as a big pattern; in the parallel mode all of you know that it goes straight, but here it will be 1 then 0, 0, 1 then 0 again 1, 0, 1, so this will be 1, 0, 0, 1, 0, 1, 0, 1.

Now this will have to be transmitted; the once we will have to be transmitted this being transmitted with respect to a particular clock, with respect to a particular clock is beat with this particular clock, but how does this system say for example, it can be bidirectional transmission, but here say this Arduino board is sending this to the PC or it

must know that some data is coming. Therefore, this usually bracketed by a start bit, stop bit.

So we will talk about this serial communication and followed by the USB communication; that is another mode of communication for this in the next class.